Future Generation Computer Systems 69 (2017) 75–87

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Bayesian network-based Virtual Machines consolidation method

Zhihua Li [a,*], Chengyu Yan [a], Xinrong Yu [a], Ning Yu [b]

[a] Department of Computer Science and Technology, School of Internet of Things Engineering, Jiangnan University, Jiangsu, Wuxi 214122, PR China

[b] Department of Computer Science, Georgia State University, GA, 30303, USA

ARTICLE INFO

Article history:
Received 8 July 2016
Received in revised form
20 November 2016
Accepted 6 December 2016
Available online 13 December 2016

Keywords:
Virtual Machines consolidation
Estimation model
Virtual Machine migration
Bayesian network

ABSTRACT

Efficient Virtual Machines (VMs) consolidation, as one of the primary methods for balancing between guaranteeing Quality of Service (QoS) and saving energy, is critical for data centers. Most existing VMs consolidation methods reallocate physical resources by adopting live VM migration. Therefore, VMs consolidation can be cast into estimating the physical resource utilization in Physical Machines (PMs) and predicting the migration probability of VMs. In this paper, we develop a Bayesian network-based estimation model (BNEM) for live VM migration, allowing a comprehensive treatment of nine actual factors in real data centers. A selection criterion of VMs to be migrated and a VM placement criterion are presented. By combining three algorithms corresponding to different phases in VMs consolidation, a hybrid Bayesian network-based VMs consolidation (BN-VMC) method is proposed. We have validated our approach by conducting a performance evaluation study using CloudSim toolkit, and the trace-driven comparison experiments are also performed. The simulation results show that the method can significantly degrade energy consumption, avoid inefficient VM migrations, and improve QoS.

1. Introduction

High energy consumption is constantly a major challenge to resource management in data centers. The scale enlargement of data centers has resulted in the high energy consumption problem [1]. Birke R et al. [2] explained that the average CPU utilization of Physical Machines (PMs) is merely 15%–20% of their common status, and most of the PMs are in an idle state. According to [3], the PMs in idle states constantly consume 70% of their peak consumption energy. Evidently, numerous idle PMs lead to low energy efficiency. Thus, as few PMs as possible should run to increase the energy efficiency of data centers.

Virtualization technology allows the creation of some Virtual Machines (VMs) in a single host and the migration of VMs to adjust the allocations in PMs. According to the resource demands of VMs, some methods [4–6] consolidated VMs into PMs to lower the load through Virtual Machine (VM) migration. In addition, VMs consolidation turns off some underloaded PMs to save energy. However, the demands of VMs may increase because the load of a data center is dynamic and uncertain. Moreover, the insignificant VMs consolidation may result in service level agreement (SLA) violations and poor Quality of Service (QoS) because of the lack of sufficient reserve resources. Performing balance between guaranteeing QoS and saving energy is one of the main challenges of dynamic VMs consolidation.

The stochastic workload negatively influences the energy consumption and performance of the system. Moreover, Voorsluys W et al. [7] emphasized that live VM migration increase the cost of computing resources, and the large scale of VM migrations may lead to extra workload, SLA violations, and considerable energy consumption. Meanwhile, the VMs suspend their service in live VM migration, and long-term migration may further affect QoS. Therefore, degrading inefficient VM migrations can enhance QoS, relieve the load imbalance of PMs, and reduce energy consumption. Apparently, we should treat when to perform VM migration and the times of VM migrations in the process of VMs consolidation.

The existing research [4–6,8] indicates that most of the VMs consolidation methods proposed only considers the case of static workload. These methods lack treatments of dynamic workload in changing data centers. Additionally, most existing VMs consolidation methods re-allocate physical resources by VM migration. To this point, we think that the essence of VMs consolidation can be cast into estimating the overload probability of PMs and predicting the probability of whether to be migrated for VMs.

Considering the aforementioned disadvantages, in this paper, our study addresses the influence of four aspects on energy

* Corresponding author.
E-mail address: zhli@jiangnan.edu.cn (Z. Li).

http://dx.doi.org/10.1016/j.future.2016.12.008
0167-739X/

consumption and QoS, namely, the dynamic workload, CPU utilization, times of VM migrations, and opportunity of VM migration from nine related factors. By associating these nine factors with a Bayesian network (BN), a BN-based estimation model (BNEM) for dynamic VM migration is created. Each issue in the different phases of VMs consolidation has been studied based on BNEM. Furthermore, by combining the individual algorithms in different phases into the corresponding process of VMs consolidation, the BN-based VMs consolidation (BN-VMC) method is proposed.

The rest of this paper is organized as follows. The related works are introduced in Section 2. In Section 3, we describe the new Bayesian network-based estimation model (BNEM) of dynamic VM migration, the function and selected reasons of each node in the BNEM are explained in detail. In Section 4, the probability of dynamic migration for VMs has been estimated based on the BNEM. By the estimation of VM migration probability, a selection criterion of the VMs to be migrated and the criterion of VM placement are conducted and proposed. Finally, we derive the framework of the BN-VMC method and present the overview and the detailed design of it. Promising experiment results are given in Section 5, and from the effectiveness and efficiency perspective, some validations and comparisons are performed, which are followed by the concluding remarks in Section 6.

2. Related works

2.1. VM placement

Research [9–12] has reasoned that the resource optimization problem in data centers is the VM placement and approximately abstracted as a bin-packing problem. Mishra M et al. [10] treated this problem to be a multi-dimensional bin-packing problem, and improved the mapping relationship between VMs and PMs with respect to the specific optimization objectives by using an improved first fit decreasing (FFD) algorithm, improved Genetic algorithm [9] and other intelligence optimization algorithms. In terms of the constraint relations between the number of VMs and PMs, the VM placement issues were classified [11], and competitive-ratio analysis was measured for an approximate solution of each type of placement problem. Although both Kaaouache M A et al. [9] and Mishra M et al. [10] focused on optimizing the usage of resources and improving the QoS by changing the mapping relation between VMs and PMs, both of them did not consider the dynamic scales of the up-allocating VMs and PMs, which easily spurred frequent VM migrations and constant on–off switches on PMs. On the other hand, the solution space of the VM placement problem can grow explosively with the increasing number of PMs and VMs, which causing difficulties in obtaining a new mapping relationship within a reasonable period of time. In contrast, VMs consolidation has to focus on which VMs need to migrate and which PMs could be turned off first, then consider the VM placement. From this perspective, the VM placement problem can be considered a sub-problem of VMs consolidation.

2.2. VMs consolidation

VMs consolidation was generally divided into the following phases [8]: host overload detection, selection of VMs to be migrated, VM placement, and running PMs shrinking. Due to the complexity of VMs consolidation, the VMs consolidation issues in [8] were separated into several sub-problems, and then the task of VMs consolidation was performed by resolving the sub-problems. Through competitive ratio-analysis, according to the viewpoint in [13], such a method is very effective in practice although this method is unable to guarantee optimal results theoretically.

Numerous researches [8,14–16] have been conducted on the issues involved in the different phases. In particular, Beloglazov A et al. [8] and Farahnakian F et al. [14] have emphasized host overload detection. The CPU overload threshold was used in [8] to maintain the CPU utilizations under static threshold after reallocating VMs; thus, considerable reserved resources can allow for workload variance, thereby ensuring QoS. Several VMs were selected from overloaded hosts and migrated to other underutilized ones. However, traditional VMs consolidation was incapable of allocating resources reasonably that incurred low performance and high energy consumption because the fixed threshold method was unable to adjust the reserved resources according to the dynamic workload. To achieve improved QoS and resource utilization when workload is variable, Farahnakian F et al. [14] proposed a dynamic VMs consolidation method based on the K-nearest neighbor (DC-KNN) algorithm. The K-nearest neighbor regression was used to predict the workload of PMs, in which the *k*-value was predicted by cross-validation, thereby migrating VMs on the basis of the cluster results. However, a long period of time was wasted when calculating the *k*-value. Similarly, the upcoming resource utilization was predicted [17] by the K-nearest neighbor, and its judgment of the overload risk of PMs was integrated with two factors including the current load and the forecast load. Further, with consideration of the uncertainty of dynamic workload in VMs consolidation of Farahnakian F et al. [17], the utilization prediction-aware best fit decreasing (UP-BFD) algorithm was presented based on a best fit decreasing (BFD) algorithm. Similar to [14], the selection of the *k*-value was unable to be adjusted quickly.

The double exponential smoothing method was used to predict the upcoming workload to PMs [15]. However, determining the parameter of the method caused by the dynamic workload was difficult. Adaptive heuristics for dynamic VMs consolidation was proposed [16]. This method adaptively adjusted the overload threshold by analyzing the historical workload trace. Three host overload detection methods were proposed [16]: median absolute deviation (MAD), interquartile range (IQR), and local regression (LR). MAD and IQR measured the workload stability by calculating MAD and IQR, respectively, of recent CPU utilizations. When carrying out unstable hosts, they decreased the overload threshold to create extra reserved resources to afford the following requirements and improve QoS. However, unstable PMs needed to reserve substantial resources for a long period of time for the cost amount of energy because MAD and IQR avoided the recent workload trends. LR predicted the CPU utilizations using local regression. Although the predicted value was able to make PMs avoid overloading, obtaining a precise prediction for the extensive value range of LR was difficult. In Farahnakian F et al. [18] and Hieu N T et al. [19], a linear regression-like method was used for predicting the CPU utilization in host overload detection. Limit look-ahead control strategy was utilized [20] to improve the mapping relationship between VMs and PMs. Because the Kalman filter method was employed to predict the forthcoming resource demands of VMs in this strategy, its performance was better in responding to the changes of dynamic workload, however, Beloglazov A et al. [8] and Han G et al. [21] have shown that the computing tasks of this strategy were too large to obtain an ideal result within a reasonable period of time. Compared with the methods [20], the proposed methods [14,16–18] were relatively simple with less computing required, thereby they were more useful and practical. To improve the energy efficiency in a data center, Masoumzadeh S et al. [22] learned the decision model by the fuzzy Q-learning technology. According to the current status of PMs, the decision model achieved a suitable overload threshold.

However, this method [22] achieved convergence after a long period of time, and was unable to adapt to the real data center. Moreover, it was difficult for each physical machine to determine the overloading without treating the recent changes of workload, and became extremely difficult in the context of the constantly changing cloud environment. Similar work was conducted in [23, 24], except that the Q-learning methods were employed in [23, 24] where the proper overload threshold and the VM migration strategies were chosen in terms of the CPU utilization of PMs and the number of VMs. Similar to the methods [22], they did not consider the recent workload changing pattern when the PMs judge the overloading. Chen L et al. [25] regarded the VM migration as a Markov decision process. To reduce the repeated migration probability of VMs, the method selected a specific VM with a particular status to migrate in terms of the PM load. Thus, the method was able to achieve load balance. However, the method lacked consideration of the perspective of energy consumption in a data center.

Beloglazov A et al. [16] also conducted research on VM migration and proposed three VM migration algorithms: minimum migration time (MMT), maximum correlation (MC), and random selection (RS). By considering the influence of live VM migrations were out of service and that the long period of time taken by the live migration of VMs further affects the QoS, the MMT preferred the VMs with a minimum migration time to migrate. However, the MMT did not effectively degrade the resource overload of PMs. Therefore, a median migration time strategy was proposed [26], which preferentially migrated the VMs with the median migration time from all the VMs in the overloaded PMs. However, there was not enough consideration of the contribution of the migrated VMs to alleviate the overloading of PMs, and the influence of the migrated VMs on the remaining VMs was not considered. Beloglazov A et al. [16] proposed the power aware best fit decreasing (PABFD) algorithm to re-deploy the migrated VMs. PABFD sorted all VMs in decreasing order based on their current CPU utilizations, and deployed each VM to a host that provided the least amount of increase in power consumption that was caused by the reallocation. However, performing load balancing of PMs caused by the algorithm preferentially allocating VMs for PMs with high-energy efficiency was difficult, thereby resulting in high loads in the PMs and poor QoS. By contrast, a few PMs had low loads and suffer from energy waste. The adaptive heuristic VMs consolidation method was also proposed [16] through combining host overload detection, selection of the VMs to be migrated, and the VM placement. However, this method only considered the CPU resources, not the times of VM migrations and when to migrate.

2.3. Intelligence-based VMs consolidation

Through combining intelligent algorithms or intelligent optimization algorithms, many researchers [27–31] have studied the VMs consolidation. Wherein, a VMs consolidation method which combines the remarked genetic algorithms (GA) was proposed [27]. Based on the treatments of the heterogeneity in PMs, the proposed method re-deployed the VMs by weighing the energy consumption and resource cost before and after VM migration in the data center. However, due to the lack of effective limitation of the scale of the participated VMs and PMs in VMs consolidation, the proposed method required a large solution space for re-deploying VMs. Moreover, the computation complexity of GA was higher than that of the FFD and BFD algorithms. So, the presented method was difficult to apply in large-scale data centers. Joshi S et al. [28] proposed a VMs consolidation algorithm based on the cuckoo algorithm. The presented method modeled the VMs consolidation issue as a multi-dimensional packing problem, and optimized a variety of resource utilizations through the model. However, the proposed method addressed the VM placement problem, and lacked constraints to VM migration and did not consider the uncertainty of the resource demand enough. Zheng Q et al. [29] proposed a VMs consolidation method based on the biogeography combining optimization algorithm. The focus of the method was at degrading the energy consumption of a data center, balancing the load, and minimizing the number of VM migrations. However, it re-deployed the VMs only depending on the current demand of resources without considering the stochastic needs. Moreover, when different VMs centrally deployed on the same PMs, it was likely to cause a high possibility of resource overload on them. Li H et al. [30] proposed a VMs consolidation method based on the particle swarm optimization (PSO) algorithm, which considered reducing energy consumption of the data center and improving resource utilization in the data center as the optimization objective. The method detected the overloaded and underloaded PMs by setting the upper and lower bounds of the static resource utilization similar to [8], which was effective in narrowing the search space of the particle swarm optimization and degrading the number of VM migrations. However, the method also avoided the uncertainty of resource demands and did not consider the possibility of VM re-migration after VM placement. Farahnakian F et al. [31] proposed a VMs consolidation method based on ant colony migration theory, which effectively degraded the number of running PMs and VM migrations. The method [31] detected the overloaded and underloaded PMs by setting the upper or lower bounds of the static resource utilization similar to [8] together with the prediction algorithm proposed in [18]. The method [31] had the advantage of degrading the search space of re-deploying VMs, which limited the number of VMs involved in VMs consolidation. However, this algorithm did not consider the probability of VM re-migration after being deployed in the process of VMs consolidation.

The proposed study conducts research on VMs consolidation with sufficient consideration of the dynamic workload, potential times of VM migrations, and when to migrate. Accordingly, we propose a BN-VMC method. First, BNEM for dynamic VM migration has been established by associating nine related factors from different views with BN, as well as with the assistance of the superior capability of probability estimation and probabilistic reasoning by such BN methods. Depending on BNEM, the migration probability of VMs deployed in different PMs with specific load patterns is estimated. The VM migration probability indicates that the potential total times of VM migrations are able to be predicted. Second, the BN-VMC method performs host overload detection by estimating the overload probability from recent workload trends, as well as adaptively adjusting the overload threshold on the basis of the overload probability, thereby resulting in a proper migration opportunity. Third, the BN-VMC method prefers the VMs with smaller memory demand and substantial effect on the potential times of VM migrations to migrate. These migrated VMs are re-deployed in the PMs with limited potential migration times and considerable available resources. Finally, the proposed method iteratively turns off the underloaded PMs to degrade energy consumption. Consequently, the BN-VMC method is capable of performing load balancing and improving QoS with limited migration times.

3. BNEM for VM migration

3.1. Assumptions of the data center

To describe the model clearly, we provide notes for the types of resources in a data center, where $H = \{h_1, h_2, \ldots, h_i, \ldots, h_n\}$ is the set of PMs and $VM_i = \left\{vm_1, vm_2, \ldots vm_j, \ldots vm_m\right\}$ is the set of VMs deployed in h_i. The CPU capacity of vm_j is denoted as c_j, r_j is the requested CPU capacity of vm_j, and d_j represents the demand of vm_j for CPU utilization, which is expressed as a percentage. Formula (1) expresses the relationship in c_j, d_j, and r_j. D_i is the accepted CPU demand of h_i and can be calculated using Formula (2). a_j is the CPU capacity that the PMs allocate to vm_j, C_i is the resource allocation of h_i, and u_i represents the CPU utilization of h_i and is defined in (3). The total amount of memory is denoted as R_j, and ram_j is the requested memory capacity of vm_j.

$$r_j = d_j \cdot c_j \tag{1}$$

$$D_i = \sum_{vm_j \in VM_i} r_j \tag{2}$$

$$u_i = \frac{1}{C_i} \sum_{vm_j \in VM_i} a_j. \tag{3}$$

3.2. Estimation model of the dynamic migration for VMs

BN is a modeling method that is effective at modeling complex uncertainty systems, as well as effectively degrading the difficulty of knowledge access and the complexity of probabilistic reasoning [32]. Fig. 1 shows BNEM estimating the probability of VM migration under sufficient treatments of the real data center, as well as according to the changing workload and other related factors in these data centers. This model can estimate the VM migration probability in different environments by using available information and probability relationships between nodes in BN. The proposed BNEM contains nine nodes. Each node represents a corresponding factor that affects the VM migration and VMs consolidation in data centers. Fig. 1 shows the relationship of each node. In the following, each of the nodes is thoroughly expressed individually, including their capability and the reason they are selected.

Node '*VM.Type*' (abbreviated as T) represents the type of VM instance; thus, different types of instances have different CPU and memory configurations. This paper is intended to prefer the VMs that occupy a relatively small amount of memory to migrate. Information about the CPU type and memory specification are needed to classify the VM instance and taken as a network node. Node '*PM.Model*' (abbreviated as M) represents type of PMs, such as HP ProLiant ML110 G4 (Intel Xeon 3040, 2 cores $\times$ 1860 MHz, 4 GB). Different specifications of PMs have different configurations of resources. After some VMs are assigned to different types of PMs, the energy consumption caused and the available computing resources are also different. Thus, the model of PMs is one of the most important factors needed by the Bayesian network. Node '*demand*' (abbreviated as d) represents the percentage of current CPU demands, whereas node '*mean*' (abbreviated as m) and '*St.dev*' (abbreviated as sd) represent the mean and standard deviation, respectively, of the recent resource demands of VMs. Node '*utilization*' (abbreviated as u) represents the CPU utilization of PMs. Node '*violate*' (abbreviated as v) represents the violations of the VM demand. For example, if $a_j < r_j$, then $v = true$, which means the demand of vm_j has not been satisfied. In general, the violations are relative to the current demands of VMs and the workloads of PMs, thereby making '*demand*', '*utilization*', '*VM. Type*', and '*PM. Model*' the parent nodes of '*violate*'.

Node '*overloaded*' (abbreviated as o) represents the overload probability of PMs. Recent research [33–37] shows that the resource demand of VM is random and can be described as stochastic models. Through deeply analyzing the trends of PlanetLab trace [38] and Google cluster trace, Yu L et al. [37] found that the workload is approximate to the normal distribution. These studies inspired us to assume the CPU demands of vm_j as $d_j \sim N\left(\mu_j, \sigma_j^2\right)$ and workload D_i of physical machine h_i is combined by the loads of all VMs in it, where μ_j and σ_j represent

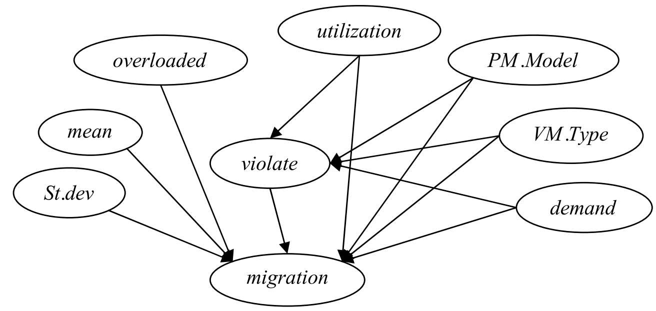


Fig. 1. BNEM for the VM dynamic migration.

the mean and the standard deviation of resource demands of VMs respectively. They can be obtained from the statistics of recently historical resource demands of vm_j; thus, $D_i \sim N\left(\mu(h_i), \sigma(h_i)^2\right)$, where $\mu(h_i)$ and $\sigma(h_i)$ can be calculated using Formulas (4) and (5), respectively. When $D_i > C_i$, the workload of h_i exceeds the CPU handling capability of the PMs; thus, this host will inevitably be overloaded. The overload probability P_{over}^i of h_i is defined in (6), where Φ is the normal probability distribution function. Nodes '*utilization*' and '*overloaded*' are used to describe the status of PMs.

$$\mu(h_i) = \sum_{vm_j \in VM_i} c_j \cdot \mu_j \tag{4}$$

$$\sigma(h_i) = \sqrt{\sum_{vm_j \in VM_i} \left(c_j \cdot \sigma_j\right)^2} \tag{5}$$

$$\begin{aligned} P_{\text{over}}^i &= \Pr(D_i > C_i) = 1 - \Pr(D_i \le C_i) \\ &= 1 - \Phi\left(\frac{C_i - \mu(h_i)}{\sigma(h_i)}\right). \end{aligned} \tag{6}$$

Node '*migration*' (abbreviated as mig) represents the VM migration. If the VM migrates, then $mig = true$; otherwise, $mig = false$. To study the probability of VM migration with different load patterns under PMs with different load status, we make the aforementioned eight nodes parent nodes of the '*migration*'.

d, m, sd, u, and o are continuous random variables; however, they must be discretized to estimate the probability in BN. The value domain of these random variables are divided into l bins, which are defined in (7) in which all the variables have the value range [0, 1].

$$\begin{aligned} &B_1 = \left[0, \frac{1}{l}\right), \qquad B_2 = \left[\frac{1}{l}, \frac{2}{l}\right), \ldots, \\ &B_b = \left[\frac{b-1}{l}, \frac{b}{l}\right), \ldots, \qquad B_l = \left[\frac{l-1}{l}, 1\right] \end{aligned} \tag{7}$$

$$f_B(x) = \frac{1}{l} \sum_{b=1}^{l} b \cdot I\left(x \in B_b\right). \tag{8}$$

Formula (8) shows the function of f_B, which can map the values to the corresponding bins, where $I(x \in B_b)$ is used to determine whether x is in the range of B_b. If yes, then set $I(x \in B_b) = 1$; otherwise, $I(x \in B_b) = 0$.

After each round of VMs consolidation, the new mapping relationship between VMs and PMs can be updated through the VMs consolidation method. At time t, according to the status of whether the VM is being migrated or not, the state of vm_j is described as the following tuple form.

$$\begin{aligned} State(vm_j, t) = &< mig_j | < v_j | f_B(d_j), f_B(u_i), T_j, M_i >, \\ &f_B(sd_j), f_B(m_j), f_B(o_i), \\ &f_B(d_j), f_B(u_i), T_j, M_i > . \end{aligned}$$

To better illustrate the above issues, an example is given as follows.

At time t, supposing that vm_j is deployed on the host h_i, the CPU utilization of h_i is 53%, the overload probability is 34%, the CPU resource demand of vm_j is 22%, the recent standard deviation and the mean of CPU resource demands are 5% and 19%, respectively. At the same time, the CPU resource demands of vm_j have not been satisfied, and the vm_j has to migrate. At this time, the status of vm_j can be described as follows.

$$\begin{aligned} State(vm_j, t) = &< mig = true| < v = true|d = 0.3, u = 0.6, \\ & T = T_j, M = M_i >, sd = 0.1, m = 0.2, \\ & o = 0.4, d = 0.3, u = 0.6, T = T_j, M = M_i > . \end{aligned}$$

The status statistic of all VMs can be conducted by the above ways. The statistics of status can be added to the observation data sets. In addition, by considering the dynamic changes of the workload in data centers, the recently generated VM status are retained in the observation data sets, and the expired status of VMs are removed.

4. BN-VMC method

4.1. BNEM-based VM migration probability estimation

According to the maximum likelihood estimation method, the parameters in the BNEM can be determined through observations.

We assume that z_i is a node of the BN and R_i is its corresponding value set. Thereafter, its parent nodes have q_i types of combinations. The condition probability of $z_i = k$ is defined in (9) according to the maximum likelihood estimation method when the status of the parent nodes is j.

$$\Pr(z_i = k|\pi(z_i) = j) = \begin{cases} \dfrac{e_{ijk}}{\sum\limits_{k \in R_i} e_{ijk}}, & \sum\limits_{k \in R_i} e_{ijk} > 0 \\ \dfrac{1}{|R_i|}, & \text{other} \end{cases} \tag{9}$$

where e_{ijk} represents the number of observation items that satisfy $z_i = k$ and $\pi(z_i) = j$. In the observations, $\sum_{k \in R_i} e_{ijk}$ is the amount of observation items that satisfy $\pi(z_i) = j$. Here, given an example for computing the condition probability of '*violate*', assuming that the CPU utilization of h_i is in [50%, 60%), the type of h_i is $M = M_1$, the CPU resource demands of vm_j is in [20%, 30%), the instance type of VMs is $T = T_1$, and $\pi(v) = (d = 0.3, T = T_1, u = 0.6, M = M_1)$. Further, the number of the statuses that satisfies $(d = 0.3, T = T_1, u = 0.6, M = M_1)$ is N_s $(N_s > 0)$, the number of the status that satisfies $(v = true|d = 0.3, u = 0.6, T = T_j, M = M_{i_1})$ is N, the violation probability of resource demands of VMs is $\Pr(v = true|d = 0.3, T = T_1, u = 0.6, M = M_1) = N/N_s$.

The migration probability P_{mig} of vm_j in h_i can be estimated, as formalized in (10).

$$\begin{aligned} & P_{\text{mig}}\left(d_j, \mu_j, \sigma_j, T_j, u_i, P^i_{\text{over}}, M_i\right) \\ & \quad = \Pr\left(v = true|f_{\pi(v)}\left(d_j, T_j, u_i, M_i\right)\right) \\ & \qquad \cdot \Pr\left(mig = true|f_{\pi(mig)}\left(v = true, d_j, \mu_j,\right.\right. \\ & \qquad \left.\left.\sigma_j, T_j, u_i, P^i_{\text{over}}, M_i\right)\right) \\ & \qquad + \Pr\left(v = false|f_{\pi(v)}\left(d_j, T_j, u_i, M_i\right)\right) \\ & \qquad \cdot \Pr\left(mig = true|f_{\pi(mig)}\left(v = false, d_j, \mu_j,\right.\right. \\ & \qquad \left.\left.\sigma_j, T_j, u_i, P^i_{\text{over}}, M_i\right)\right) \end{aligned} \tag{10}$$

where $d_j, \mu_j, \sigma_j, T_j$ represent the resource demands, recent resource demands, standard deviation, and type of VM instance of vm_j, respectively; and u_i, P^i_{over}, M_i represent the resource utilization, overload probability, and type of PM h_i, respectively.

Formula (11) shows that function $f_{\pi(v)}$ maps the observations of parent nodes $\pi(v) = \{d, u, T, M\}$ to the corresponding interval. Function $f_{\pi(mig)}$ works in the same way as that in $f_{\pi(v)}$.

$$\begin{aligned} f_{\pi(v)}\left(d_j, T_j, u_i, M_i\right) = \big(& d = f_B(d_j), \\ & T = T_j, u = f_B(u_i), M = M_i\big). \end{aligned} \tag{11}$$

4.2. Adaptive host overload detection

The current resource utilization of PMs can often reflect the recent workload level. The overload probability can measure the potential overload risk of PMs with the current load pattern. When the overload probability of a physical machine is very high, it indicates that the resource demands of VMs deployed on it are more likely not to be satisfied. Therefore, some of the deployed VMs need to migrate to degrade the loads on the PMs, so the QoS is not affected. Given that the workload in a data center is dynamic, the estimation of the overload probability EOP algorithm was proposed to detect the overloaded PMs by considering the current resource utilization and potential overload probability in the current PMs. This estimation aims to detect the PMs with the overload risk. First, the EOP algorithm provides two following assumptions. (1) When the overload probability is 0, the threshold is 100% and does not migrate any VMs. (2) When the overload probability is 100%, the PMs will not be overloaded but only if the utilization is under certain specific value. Thus, the overload threshold T^i_{u} of host h_i can be defined in (12).

$$T^i_{\text{u}} = 1 - s \times P^i_{\text{over}} \tag{12}$$

where parameter s represents the trustworthy level of the overload probability and can weigh the relationship between resource utilization of PMs and QoS. When s increases, T^i_{u} is so sensitive to the changes in P^i_{over} that the selected T^i_{u} relatively decreases, that is, T^i_{u} decreases for an increase in P^i_{over}. As a result, it causes the PMs to save more resources after each round of VMs consolidation and has the benefit of guaranteeing QoS. Conversely, while s decreases, the trustworthiness of the overload probability degrades, T^i_{u} is so insensitive to the changes in P^i_{over} that the selected T^i_{u} increases. If so, it is inclined to maximize the utilization of resource in PMs, helping to save energy.

As for host overload detection, if the CPU utilization of a host is $u_i > T^i_{\text{u}}$, then the host h_i is overloaded. Thus, the EOP algorithm is shown as Algorithm 1.

Algorithm 1: Estimation of the overload probability

```
Input: h_i
Output: isOverloaded_i
Use (6) calculate P^i_over
Use (12) calculate T^i_u
if u_i > T^i_u then
    isOverloaded_i ← true
else
    isOverloaded_i ← false
end if
```

4.3. Selection of the VMs to be migrated

When a host is identified overload, some VMs have to migrate to improve the QoS. The VMs need to suspend their service while migrating themselves. To degrade the time of VM migration, VMs with smaller memory capacity are preferred.

Along with each round of VM migration, the CPU utilization and overload probability of the PMs will change, and the probability of the remaining VMs to be migrated in PMs will be changed as well. Based on BNEM, the total migration times Mig_i^{-k} of the remaining VMs can be estimated using Formula (13) after the host h_i migrates the VM vm_k. Accordingly, u_i^{-k} represents the CPU utilization as described in (14) and P_{over}^{i-k} represents the overload probability after h_i migrates vm_k, which is calculated using (15). The average current demands of resource $\mu^{-k}(h_i)$ and its standard deviation $\sigma^{-k}(h_i)$ are defined in (16) and (17), respectively.

$$Mig_i^{-k} = \sum_{vm_j \in VM_i - \{vm_k\}} P_{\text{mig}}\left(d_j, \mu_j, \sigma_j, T_j, u_i^{-k}, P_{\text{over}}^{i-k}, M_i\right) \cdot 1 \tag{13}$$

$$u_i^{-k} = \frac{1}{C_i} \sum_{vm_j \in VM_i - \{vm_k\}} a_j \tag{14}$$

$$P_{\text{over}}^{i-k} = 1 - \Phi\left(\frac{C_i - \mu^{-k}(h_i)}{\sigma^{-k}(h_i)}\right) \tag{15}$$

$$\mu^{-k}(h_i) = \sum_{vm_j \in VM_i - \{vm_k\}} c_j \cdot \mu_j \tag{16}$$

$$\sigma^{-k}(h_i) = \sqrt{\sum_{vm_j \in VM_i - \{vm_k\}} \left(c_j \cdot \sigma_j\right)^2}. \tag{17}$$

To degrade the possibility of subsequent VM migrations, VMs with low Mig_i^{-k} have priority for migration.

Under the premise of considering the total times and opportunity of VM migrations, a presented selection criterion of VMs to be migrated is defined in (18).

$$g_M^{i-k} = \frac{ram_k}{net_i} + \alpha \cdot Mig_i^{-k} \tag{18}$$

where ram_k represents the memory capacity occupied by vm_k, net_i is the bandwidth provided by h_i, and α is the weight of Mig_i^{-k} related to migration times. Based on the presented selection criterion of VMs to be migrated, the migration and capacity aware migration selection (MCAMS) algorithm is proposed (see Algorithm 2).

Algorithm 2: Migration and capacity aware migration selection

Input: h_i
Output: VM_m
$VM_m \leftarrow \varnothing$
while $EOP(h_i) = true$ **do**
  $vm_{\text{mig}} \leftarrow \varnothing$
  $g_{\min} \leftarrow MAX$
  for all $vm_k \in VM_i$ **do**
    Use (18) calculate g_M^{i-k}
    if $g_M^{i-k} < g_{\min}$ **then**
      $g_{\min} \leftarrow g_M^{i-k}$
      $vm_{\text{mig}} \leftarrow vm_k$
    end if
  end for
  $VM_m \leftarrow VM_m \cup \{vm_{\text{mig}}\}$
  $VM_i \leftarrow VM_i - \{vm_{\text{mig}}\}$
end while

4.4. VM placement

After each host accepts a new VM, its load status likely change and the migration probability of each VM on this physical machine synchronously vary.

BNEM states that after vm_k is migrated to h_i, the total possible migrations Mig_i^{+k} of all VMs deployed on h_i can be estimated using (19). After VM migration, u_i^{+k} represents the CPU utilization and is defined using (20). P_{over}^{i+k} represents the overload probability and is defined in (21). The recent resource demands $\mu^{+k}(h_i)$ and the standard deviation $\sigma^{+k}(h_i)$ can be calculated using (22) and (23), respectively. Lower Mig_i^{+k} values represent h_i as more fit to vm_k, and the probability of migration caused by the resource overload will decrease. The lower the Mig_i^{+k} value is, the less likely VM migration events caused by resource overload are to occur, thereby indicating that the host h_i and demand of vm_k are substantially consistent.

$$Mig_i^{+k} = \sum_{vm_j \in VM_i \bigcup \{vm_k\}} P_{\text{mig}}\left(d_j, \mu_j, \sigma_j, T_j, u_i^{+k}, P_{\text{over}}^{i+k}, M_i\right) \cdot 1 \tag{19}$$

$$u_i^{+k} = \frac{1}{C_i} \sum_{vm_j \in VM_i \bigcup \{vm_k\}} a_j \tag{20}$$

$$P_{\text{over}}^{i+k} = 1 - \Phi\left(\frac{C_i - \mu^{+k}(h_i)}{\sigma^{+k}(h_i)}\right) \tag{21}$$

$$\mu^{+k}(h_i) = \sum_{vm_j \in VM_i \bigcup \{vm_k\}} c_j \cdot \mu_j \tag{22}$$

$$\sigma^{+k}(h_i) = \sqrt{\sum_{vm_j \in VM_i \bigcup \{vm_k\}} \left(c_j \cdot \sigma_j\right)^2}. \tag{23}$$

In terms of the comprehensive consideration of energy consumption and VM migrations, a presented VM placement criterion is defined in (24).

$$g_A^{i+k} = \left(p_i^{\text{p}} - p_i^{\text{i}}\right) \cdot (1 - u_i) - \beta \cdot Mig_i^{+k} \tag{24}$$

where p_i^{p} represents the peak power of h_i, p_i^{i} represents idle power, u_i represents the current CPU utilization, and β represents the weight of Mig_i^{+k} related to the effective remainder energy.

A migration and power aware best fit decreasing (MPABFD) algorithm is proposed based on the presented VM placement criterion (see Algorithm 3).

Algorithm 3: Migration and power aware best fit decreasing

Input: VM_m, H_a
Sort VM_m **in decreasing order of request CPU capacity**
for all $vm_k \in VM_m$ **do**
  $g_{\max} \leftarrow MIN$
  $h_t \leftarrow \varnothing$
  for all $h_i \in H_a$ **do**
    if h_i *has enough resources for* vm_k **then**
      $VM_i \leftarrow VM_i \cup \{vm_k\}$
      if $EOP(h_i) = false$ **then**
        $VM_i \leftarrow VM_i - \{vm_k\}$
        Use (20) to calculate g_M^{i+k}.
        if $g_M^{i+k} > g_{\max}$ **then**
          $g_{\max} \leftarrow g_M^{i+k}$
          $h_t \leftarrow h_i$
        end if
      else
        $VM_i \leftarrow VM_i - \{vm_k\}$
      end if
    end if
  end for
  if $h_t \neq \varnothing$ **then**
    $VM_t \leftarrow VM_t \cup \{vm_k\}$
    $VM_m \leftarrow VM_m - \{vm_k\}$
  end if
end for

4.5. BN-based VMs consolidation

By combining the EOP, MCAMS, and MPABFD algorithms in terms of their inner relationship, the BN-VMC method has been proposed, which includes four phases. The first phase uses EOP and determines the PMs that have the overload risks. The second phase uses MCAMS to migrate some VMs from the PMs with overload risks, as well as enables the overload alert condition to not be satisfied. The third phase re-deploys the migrated VMs to the new destination PMs using MPABFD. Finally, contract the running PMs and migrate all VMs to underloaded ones and switch these hosts to sleep to decrease energy consumption. To turn off more hosts, the BN-VMC algorithm utilizes the iteration strategy, thereby switching PMs with the lowest resource utilization to sleep. Additionally, in real data centers, VMs consolidation is carried out for one round in a fixed period of time. Hence, the BN-VMC method periodically conducts VMs consolidation. The BN-VMC method has the potential to improve QoS and enhance resource utilization based on the comprehensive analysis of the separated algorithm (e.g., EOP, MCAMS, and MPABFD). The BN-VMC method is described as Algorithm 4.

Algorithm 4: BN-VMC method

for all $h_i \in H_a$ **do**
  if $EOP(h_i) = true$ **then**
    $H_o \leftarrow H_o \cup \{h_i\}$
    $VM_m \leftarrow VM_m \cup MCAMS(h_i)$
  end if
end for
$MPABFD(VM_m, H_a - H_o)$
$H_a \leftarrow H_a - H_o$
while $H_a \neq \varnothing$ **do**
  $h_{\min} \leftarrow \arg\min_{h_i \in H_a} u_i$
  $MPABFD(VM_{\min}, H_a - \{h_{\min}\})$
  if $VM_{\min} \neq \varnothing$ **then**
    break
  else
    $H_a \leftarrow H_a - \{h_{\min}\}$
  end if
end while

5. Implementation and simulation

5.1. Experiment environment

This study employs CloudSim toolkit [39] as the simulation platform. The experiments simulate a data center comprised of 800 heterogeneous PMs. These PMs are classified into two categories: HP ProLiant ML110 G4 (Intel Xeon 3040 2 Cores 1860 MHz, 4 GB) and HP ProLiant ML110 G5 (Intel Xeon 3075 2 Cores 2260 MHz, 4 GB). Four types of VMs are used in the presented experiments: High-CPU Medium Instance (2500 MIPS, 0.85 GB), Extra Large Instance (2000 MIPS, 3.75 GB), Small Instance (1000 MIPS, 1.7 GB), and Micro Instance (500 MIPS, 613 MB). After creating PM instances and VM instances on the CloudSim platform, the VMs are deployed to different PMs in a round-robin manner. Because the total number of created VMs is greater than the PMs, all 800 PMs are running after initialization. The approach is able to evaluate the capability of such methods on VMs consolidation. After each round of VMs consolidation, the workload traces and resource demands of VM changes. In this paper, we use two traces: PlanetLab [16] trace and Bitbrains trace [40]. The PlanetLab trace records the CPU utilization of VMs in PlanetLab platform every five minutes in 10 random days in March and April 2011. Table 1 shows the related information. Bitbrains releases a data set containing workload trace for altogether 1750 VMs from its hosting center, representing business-critical enterprise applications. The trace is comprised of one file per VM, describing mainly the VM's dynamic workload, sampled every 5 min. Table 2 shows its properties.

Table 1
PlanetLab trace properties (CPU utilization).

Date	Number of VMs	Mean (%)	St. dev. (%)
03/03/2011	1052	12.31	17.09
06/03/2011	898	11.44	16.83
09/03/2011	1061	10.70	15.57
22/03/2011	1516	9.26	12.78
25/03/2011	1078	10.56	14.14
03/04/2011	1463	12.39	16.55
09/04/2011	1358	11.12	15.09
11/04/2011	1233	11.56	15.07
12/04/2011	1054	11.54	15.15
20/04/2011	1033	10.43	15.21

Table 2
Bitbrains trace properties (CPU utilization).

Date	Number of VMs	Mean (%)	St. dev. (%)
2013-08-01	1238	10.86	5.91
2013-08-02	1237	7.28	6.07
2013-08-03	1234	4.78	3.79
2013-08-04	1233	8.15	4.90
2013-08-05	1232	9.11	6.48
2013-08-06	1231	8.31	3.88
2013-08-07	1218	7.40	6.45
2013-08-08	1209	10.43	6.77
2013-08-09	1207	6.72	6.00
2013-08-10	1205	8.29	5.53

5.2. Evaluation indices

To reasonably evaluate the algorithm performance, the presented study adopts six indices that were proposed in [16]: service level agreement (SLA) violation time per active host (SLATAH), performance degradation due to migrations (PDM), SLA violations (SLAV), energy consumption (EC), VM migrations (VMM), and energy and SLA violations (ESV).

SLATAH is defined in (25), where T_i^{s} is the time of SLAV caused by the CPU resource overload of h_i, T_i^{a} is the running time of h_i, and n is the total number of hosts. PDM is described using (26), where C_j^{d} is the unsatisfied CPU required capacity caused by the migrating vm_j, C_j^{r} is the CPU capacity required by vm_j, and m represents the total number of VMs. SLATAH measures QoS provided by the running PMs and PDM measures the performance degradation of VMs caused by migration. SLAV measures the single-day QoS of the data center through the aforementioned two indices, as described in (27). The considerably low three indices, SLATAH, PDM, and SLAV, indicate a high QoS.

$$SLATAH = \frac{1}{n}\sum_{i=1}^{n}\frac{T_i^{\mathrm{s}}}{T_i^{\mathrm{a}}} \tag{25}$$

$$PDM = \frac{1}{m}\sum_{j=1}^{m}\frac{C_j^{\mathrm{d}}}{C_j^{\mathrm{r}}} \tag{26}$$

$$SLAV = SLATAH \times PDM. \tag{27}$$

EC represents the energy consumption of a data center per day. A considerably low EC means high energy efficiency and limited energy consumption. ESV is defined in (28). A low estimation of ESV indicates improved EC performance and QoS in the data center.

$$ESV = EC \times SLAV. \tag{28}$$

While a VM is migrating, it usually suspends its service, thereby likely influencing its QoS. Thus, degrading the inefficient VM migrations enhances QoS. Therefore, less VM migrations represent a high performance of the VMs consolidation.

5.3. Results analysis

In this section, several experiments are arranged to validate the effectiveness and efficiency of the proposed method from different aspects.

1. Effectiveness

To validate the utility of the proposed method, the dynamic VMs consolidation approaches proposed in Beloglazov A et al. [8] and Beloglazov A et al. [16], including four host overload detection algorithms (i.e., ST, MAD, IQR, and LR) and three VMs selection strategies (i.e., MMT, MC, and RS), are implemented on the CloudSim platform. Based on [8,16], the parameters for the overload detection algorithms are set as follows: MAD-2.5, IQR-1.5, LR-1.2, and ST-0.8, in which the latter numeric value after the algorithm name is the corresponding parameter in the corresponding algorithms. The parameter s of BN-VMC is 1.0, α is 0.4, and β is 0.4. The experiments involved in the BN-VMC method are compared with 12 other combinations of the overload detection and VM selection algorithms, as well as UP-BFD and DC-KNN algorithms. Tables 3 and 4 show the experiment results, in which the numeric value behind each algorithm name (i.e., MAD-MMT-2.5, LR-MMT-1.2, and IQR-MMT-1.5, etc.) represents their related parameter in the corresponding algorithms.

EC index can be utilized to better measure the energy consumption of different methods. From Tables 3 and 4, the BN-VMC method has the least amount of energy consumption among all the compared methods using Planetlab and Bitbrains trace. However, the UP-BFD, LR-MC-1.2, and LR-RS-1.2 also show good performance, but the energy consumption of these methods is greater than that of the BN-VMC method. Furthermore, the energy consumption of the methods that contain MMT algorithm is greater than the energy consumption of the methods containing MC or RS algorithms. On the Planetlab and Bitbrains trace, the energy consumption of ST-MMT-0.8 is 188.5 and 90.36, respectively, which is the largest one among all the compared algorithms. This is due to the MMT algorithm migrating more insignificant VMs, which results in the degraded energy efficiency. In addition, the dynamic uncertainty of workload in a data center is not considered in the overload threshold selection policy of the ST algorithm. As a result, the selected overload threshold is so unreasonable that the energy consumption increases. The BN-VMC method outperforms the other methods in terms of saving energy. This is primarily because the BN-VMC method has the priority to select the underloaded PMs during the VM placement and reasonable overload threshold. This strategy is very effective in degrading energy consumption.

The SLAV index mainly reflects the capability of different algorithms in guaranteeing QoS. As shown in Tables 3 and 4, the SLAV value of BN-VMC is the smallest among all the compared algorithms, and the SLAV value of UP-BFD is the second smallest. Using Planetlab, the SLAV value of LR-RS-1.2 is the largest. Using Bitbrains trace, the SLAV value of MAD-RS-2.5 is the largest. It can be found that this situation is caused by the RS algorithm, in which, the VMs to be migrated are randomly selected. This strategy is too simple to improve the QoS and degrade energy consumption. The capability of the methods with RS algorithms in guaranteeing service quality is poor.

Because too many VM migrations can increase energy consumption, the times of VM migrations reflects the energy cost and QoS from another aspect. On the two workload traces including Planetlab and Bitbrains, the times of VM migrations triggered by BN-VMC are the smallest among all the compared algorithms, and the times of the UP-BFD algorithm is the second smallest. This is primarily because the BNEM model inside the BN-VMC method considers diverse factors and the selected factors are reasonable and effective. In addition, as shown in Tables 3 and 4, the times of VM migration in the methods that combine the MMT algorithm are typically more than those of other compared algorithms. This is because no consideration is given to the influence of VM migrations to the load balancing of destination PMs when the MMT algorithm selects the up-migrating VMs. Consequently, the BN-VMC method can effectively degrade the times of VM migrations.

Table 3
Simulation results using Planetlab trace with four indices.

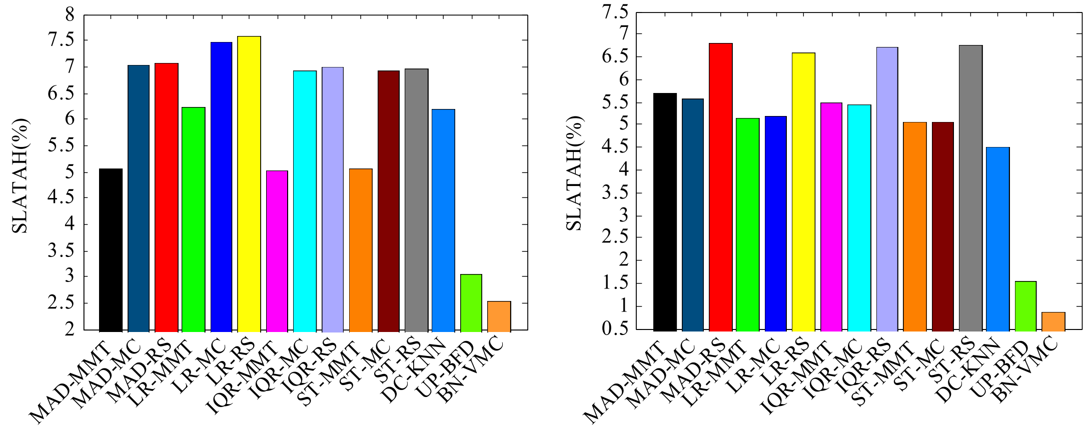
Method	EC (kWh)	SLAV	VMMs	ESV (%)
BN-VMC	109.70	0.001422	15 827	0.1560
UP-BFD	133.10	0.002345	22 090	0.3086
DC-KNN	152.57	0.004530	25 708	0.6802
MAD-MMT-2.5	183.50	0.003348	26 305	0.6143
MAD-MC-2.5	173.79	0.007111	23 420	1.2358
MAD-RS-2.5	174.86	0.007038	23 642	1.2307
LR-MMT-1.2	161.87	0.004974	28 175	0.8051
LR-MC-1.2	148.51	0.007609	23 931	1.1230
LR-RS-1.2	147.67	0.007807	23 659	1.1528
IQR-MMT-1.5	187.53	0.003288	26 497	0.6166
IQR-MC-1.5	177.70	0.006805	23 394	1.2093
IQR-RS-1.5	178.61	0.006865	23 740	1.2262
ST-MMT-0.8	188.50	0.003371	26 602	0.6354
ST-MC-0.8	179.38	0.006989	23 962	1.2537
ST-RS-0.8	180.43	0.006970	24 155	1.2576

Table 4
Simulation results using Bitbrains trace with four indices.

Method	EC (kWh)	SLAV	VMMs	ESV (%)
BN-VMC	52.68	0.000188	5 181	0.0075
UP-BFD	63.24	0.000471	8 091	0.0242
DC-KNN	80.14	0.002016	12 674	0.1476
MAD-MMT-2.5	84.39	0.003962	18 510	0.3492
MAD-MC-2.5	83.59	0.003758	17 727	0.3242
MAD-RS-2.5	80.56	0.004572	14 948	0.3655
LR-MMT-1.2	81.29	0.002686	13 861	0.2181
LR-MC-1.2	81.11	0.002711	13 845	0.2191
LR-RS-1.2	77.29	0.003745	12 240	0.2820
IQR-MMT-1.5	85.57	0.003650	17 508	0.3229
IQR-MC-1.5	85.45	0.003586	16 959	0.3167
IQR-RS-1.5	82.89	0.004395	14 362	0.3615
ST-MMT-0.8	90.36	0.002904	15 128	0.2692
ST-MC-0.8	89.96	0.002890	14 987	0.2668
ST-RS-0.8	87.72	0.004344	13 854	0.3859

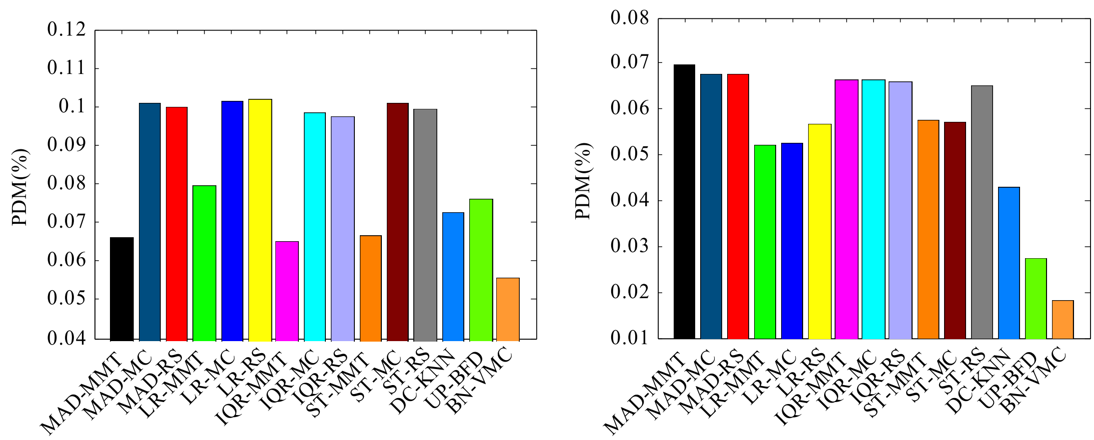
Figs. 2 and 3 show further analyses of SLATAH and PDM, respectively. In Figs. 2 and 3, different colored histograms indicate different combination methods, with the vertical axis representing the SLATAH or PDM value of the compared algorithms, which are expressed as a percentage. Fig. 2 shows the results of the compared methods in the SLATAH index, in which BN-VMC can ensure the QoS of running PMs. Meanwhile, the overload probability of the PMs decreased. Fig. 3 shows the results of the compared methods in PDM, in which BN-VMC also efficiently decreases the occurring probability of influencing QoS caused by the live migration of VMs suspending their service. This effect is a result of extra-inefficient VM migration being limited in the BN-VMC method. The ideal opportunity for VM migration is achieved, thereby degrading the insignificant VM migration, shortening the break time of VM migration, and decreasing the risk of overloaded occurrences in the PMs. This effect is considered to be promising. The same conclusion can be achieved from the conjunction perspective of Figs. 2, 3, Tables 3 and 4, particularly the VMMs index (i.e., the times of VM migrations) in Tables 3 and 4. Thus, the BN-VMC method is more capable than the other compared methods in terms of the SLAV index and PDM index.

ESV index is a comprehensive measurement for evaluation of the energy consumption and QoS. From Tables 3 and 4, the ESV



(a) Experimental results using PlanetLab trace. (b) Experimental results using Bitbrains trace.

Fig. 2. Comparison of SLATAH. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) Experimental results using PlanetLab trace. (b) Experimental results using Bitbrains trace.

Fig. 3. Comparison of PDM. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

index of BN-VMC is the minimal one and is lower than that of the other algorithms, indicating promising performance.

In summary, the proposed BN-VMC method shows the best performance in terms of EC, SLAV, VMM, and ESV. This method consistently achieves an improved balance between energy consumption and guaranteeing QoS, as well as in enhancing energy efficiency while degrading inefficient VM migrations.

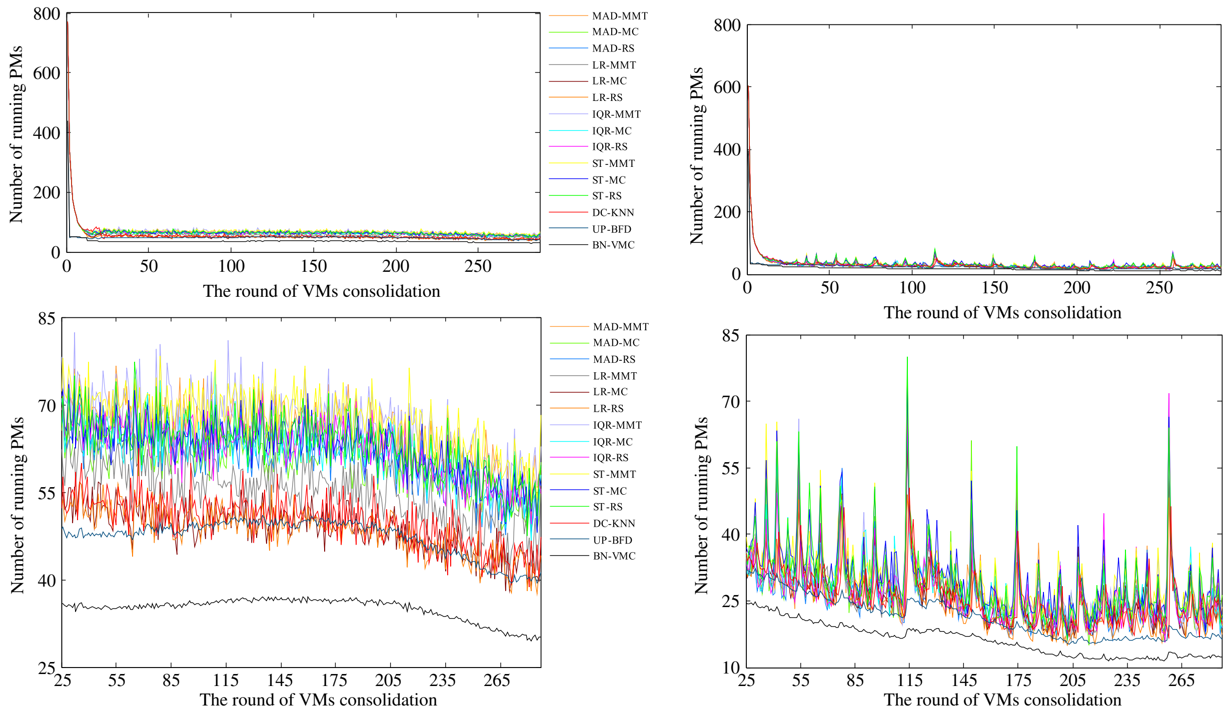
2. Efficiency

The definition of ESV index, as shown in (28) in Section 5.2, is a comprehensive evaluation index about the ideal balancing between energy consumption and guaranteeing QoS. The following study further analyze the BN-VMC method. In-depth experimental analyses were performed between the BN-VMC method and the other methods. In a real data center, because the actual workload is regular and periodic, we use a one-day workload as the experiment traces. Here, to conduct a round of VMs consolidation every 5 min, the total number of rounds is 288. Figs. 4–7 shows the results.

Figs. 4 and 5 each have two sub-graphs including (a) and (b). Although a different sub-graph corresponds to different experiment traces including PlanetLab and Bitbrains, their infrastructures are similar. Each sub-graph is partitioned into upper and lower parts. In each sub-graph, the upper graph shows the results from the start of VMs consolidation to the 288th round of VMs consolidation. To enhance the distinguished divisions for the compared algorithms, the lower part of a sub-graph is a part of the upper graph, including the results from the 25th to the 288th round. The different colored lines indicate the different compared methods, which are indicated on the right side of each sub-graph. The correspondence is the same in different sub-graphs. In the following figures, the coordinate value "0" shows the start of VMs consolidation.

Fig. 4 shows that the number of running PMs varied while the VMs consolidation continues to be performed. Fig. 4(a) shows the experimental results using Planetlab trace. The upper sub-graph of Fig. 4(a) shows the changing trend between the number of PMs and VMs consolidation during the first round of VMs consolidation to the 288th round. The BN-VMC method is able to turn off a large number of PMs more quickly than the other algorithms at initial periods and decrease more energy consumption. The lower sub-graph of Fig. 4(a) shows the varying tendency between the number of PMs and VMs consolidation during the 25th round of VMs consolidation to the 288th round. The BN-VMC algorithm can maintain the number of running PMs at approximately 30, and the UP-BFD algorithm maintain the number of running PMs at approximately 45, so the energy consumption caused by the BN-VMC is lower than that of UP-BFD and much lower than the other compared algorithms. Fig. 4(b) shows the experimental results on Bitbrains trace. The results are similar to those in Fig. 4(a). As a summary, the BN-VMC algorithm can efficiently turn off a large number of PMs, and the total number of running PMs is maintained at approximately 15. Compared with the other algorithms, the BN-VMC method is superior for degrading energy consumption.

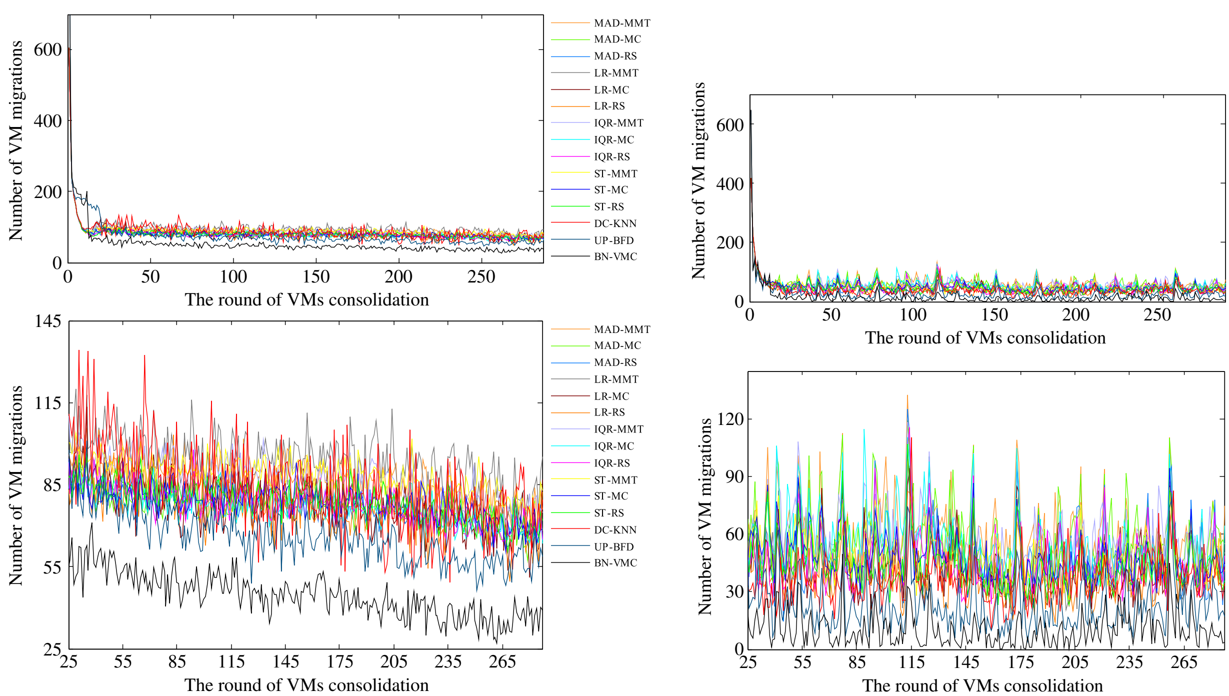
Fig. 5 shows that the constant changes on the times of the triggered VM migrations during each round vary with VMs consolidation. Fig. 5(a) is the experimental results using Planetlab trace, in which the upper sub-graph shows the changes in



(a) Experimental results using PlanetLab trace.

(b) Experimental results using Bitbrains trace.

Fig. 4. Number of the running PMs varying with the round of VMs consolidation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) Experimental results using PlanetLab trace.

(b) Experimental results using Bitbrains trace.

Fig. 5. Number of VM migrations varying with the round of VMs consolidation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

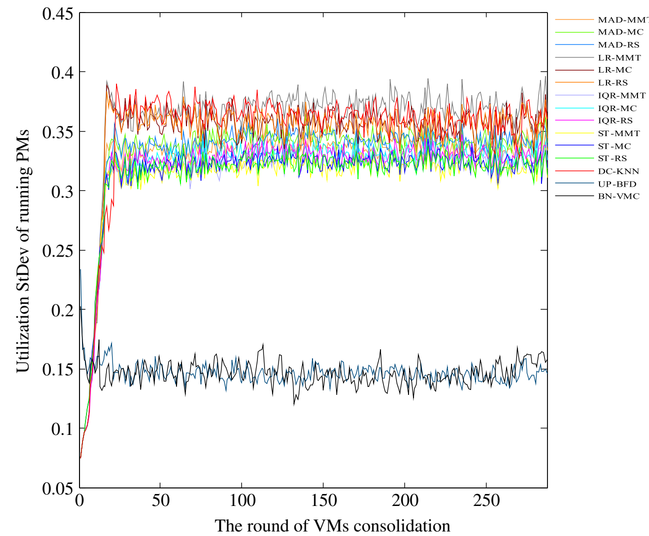
migration times during the first round of VMs consolidation to the 288th round. Compared with the other algorithms, BN-VMC triggers VM migration more frequently in the former 25 rounds. With the combined analysis shown in Fig. 4(a), it is mainly because the BN-VMC method turns off numerous PMs, resulting in a large number of VMs migrated. Further, as shown in Fig. 5(a), after the 25th round of VMs consolidation, in each round, the times of VM migrations remains at approximately 45. The times of VM migrations are significantly less than the other algorithms due to the probability of VM migration being significantly degraded after adjusting the mapping relationship by the BN-VMC method. These results show that the VM migration selection strategy and the VM placement algorithm based on BNEM model are effective. Similarly, Fig. 5(b) shows the experimental results on

Bitbrains trace, although the times of VM migrations have obvious fluctuations, the times of VM migrations triggered by the BN-VMC method at each round of VMs consolidation is the minimum among all the compared algorithms, remaining at approximately 15.

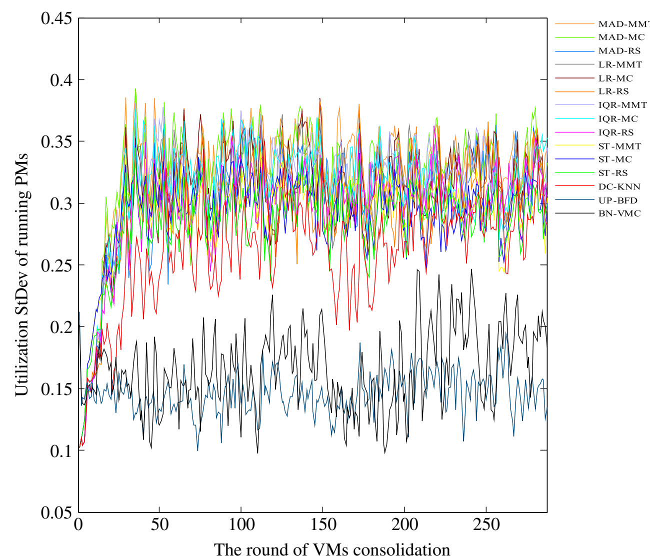
Further, by calculating the standard deviation of current CPU utilization of all running PMs, the standard deviation (abbreviated as StDev) indicates the load balancing level of the data center. The load allocation within the data center is more balancing for smaller standard deviations. Otherwise, it shows the imbalance allocation. Fig. 6 shows the level of load balancing distribution in the data center at each round along with the ongoing VMs consolidation. In Fig. 6, the different colored lines correspond to different algorithms. The corresponding relation is shown beside the sub-graph. Fig. 6(a) shows the experimental results using Planetlab. The standard deviation originated by BN-VMC and UP-BFD algorithm remains stable at approximately 0.15. As the manner of VM deployment in the experiments is evenly rotated, the initial load distribution is balancing. The standard deviation caused by the BN-VMC method is greater than that of the other algorithms. The reason for this situation is the rapid closure of a large number of PMs at the early stage, resulting in an imbalance distribution. With the ongoing VMs consolidation, after a large number of PMs are turned off, BN-VMC is still in a balancing state, while the other algorithms become imbalance. Here, we take PABFD and DC-KNN algorithm as examples to indicate the reasons that cause this imbalance phenomenon. The deployment strategy of VMs gives priority to PMs with high energy efficiency in PABFD and DC-KNN algorithms. The deployment strategy likely causes the situation that although some underloaded PMs do not turn off against they are still not fully utilized. Consequently, this results in the emergence of the load imbalance. Fig. 6(b) shows the experimental results on Bitbrains trace. Although the results are more volatile, the standard deviation of BN-VMC and UP-BFD remain stable at approximately 0.15. The above results validate that the BN-VMC and UP-BFD method can assign workload balancedly.

CPU utilization is a benchmark index that reflects the level of the efficient usage of resources. Fig. 7 shows the average of CPU utilization along with the ongoing VMs consolidation at each round in the data center. In Fig. 7, the different colored lines correspond to different algorithms. The corresponding relation is shown beside the sub-graphs. Fig. 7(a) shows the experimental results using Planetlab trace. The average CPU utilization of the BN-VMC and UP-BFD algorithms increases to greater than 70% after several rounds of VMs consolidation. Moreover, the average CPU utilization of the BN-VMC algorithm can be stabilized at 80%, significantly greater than that of the other algorithms. Fig. 7(b) shows the experimental results using Bitbrains trace. The changing trend of CPU utilization is similar to the trend in Fig. 7(a). In summary, these experimental results indicate that the BN-VMC algorithm has the capability to effectively use the CPU resources.

In Figs. 4–7, the proposed BN-VMC method outperforms the compared methods besides the utilization StDev of running PMs index. Since the UP-BFD algorithm [17] likes the BN-VMC perfectly treat the stochastic workload in VMs consolidation, their StDev of running PMs index are similar. The other evaluation indices of UP-BFD are inferior to the BN-VMC due to its employed K-nearest neighbor cluster method not to achieve a global optimization of resource demands. In general, the BN-VMC method is reliable with relatively high efficiency and can reasonably utilize computing resources, avoid extra insignificant VM migrations, guarantee good QoS, and limit inefficient resource consumption.



(a) Experimental results using PlanetLab trace.



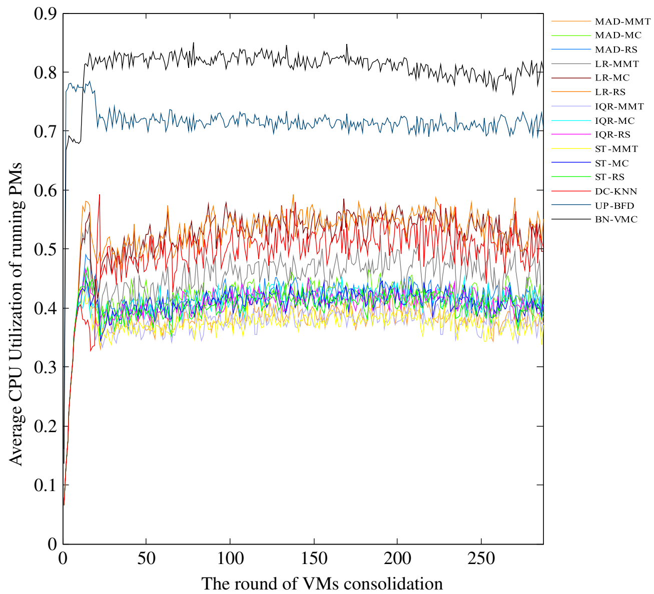
(b) Experimental results using Bitbrains trace.

Fig. 6. Standard deviation of the CPU utilization of running PMs varying with the round of VMs consolidation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

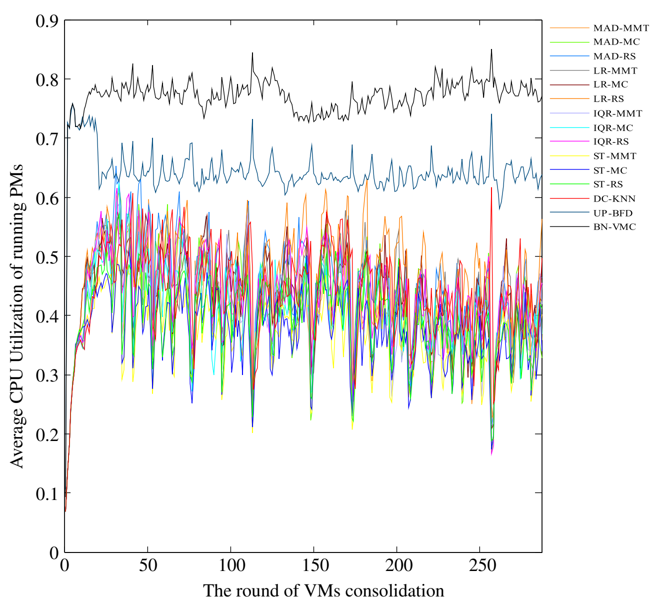
6. Conclusions

In this paper, the problem of VMs consolidation based on Bayesian network was studied and validated. This issue of VMs consolidation has received attention by researchers in recent years due to the contradiction between saving energy and guaranteeing QoS that exists in real data centers.

Three main contributions were made in this paper. First, with sufficient consideration of several factors in a real data center, including the dynamic workload, number of VM migrations, opportunity for VM migration, and CPU utilization, among others, and taking these factors as specific and parametric, these factors are equivalent to nine nodes. After nine nodes are linked with a Bayesian network, a BNEM for dynamic VM migration is created. Each node represents one aspect of VM migration that must be considered, including the recent VM resource demands, violation of the demands of VMs, CPU resource demands, etc. Second, the VM migration probability is successfully estimated according to the occurrence probability of each affair and their relationship



(a) Experimental results using PlanetLab trace.



(b) Experimental results using Bitbrains trace.

Fig. 7. Average CPU utilization of the running PMs varying with the round of VMs consolidation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

among the different factors by the BNEM. Furthermore, the issues of VMs consolidation based on BNEM (e.g., adaptive host overload detection, VM migration selection, and VM placement methods) are studied. The corresponding algorithms are also presented. Finally, the BN-VMC method is proposed by combining the aforementioned individual algorithms into corresponding phases in VMs consolidation. The proposed method determines the migration opportunity of VMs by BNEM which avoids the extra insignificant VM migration and limits inefficient VM migration. Since Bayesian network is effective at accessing priori knowledge and probability reasoning, the BNEM and BN-VMC method are suitable for VMs consolidation in changing data centers.

The trace-driven experiments are used to validate the proposed method and the experimental results show that it significantly degrades energy consumption, avoids extra insignificant VM migrations, and improves QoS. Objectively, the BN-VMC method can reasonably implement the computing resources and degrade extra inefficient resource consumption. To this point, BN-VMC is a promising and reliable method.

Although, in the process of VM migration and VMs consolidation, the demand of CPU resources is random, and the CPU resources are one of the main factors that affects energy consumption. In this paper, we fully consider the aforementioned issue. Certainly, other resource demands, such as disk usage, network load, and so on, are also random and affect the energy consumption and QoS in a data center. To degrade the energy consumption and guarantee QoS of a data center, predicting the changes and optimizing the coordination and cooperation relations is very challenging and remains a subject of future works.

Acknowledgments

This work is supported by the Future Research Projects Funds for the Science and Technology Department of Jiangsu Province (Grant No. BY2013015-23) and the Fundamental Research Funds for the Ministry of Education (Grant No. JUSRP211A 41).

Appendix A. Supplementary data

Supplementary material related to this article can be found online at http://dx.doi.org/10.1016/j.future.2016.12.008.

References

[1] M.F. Li, J.P. Bi, Z.C. Li, Resource-scheduling-waiting-aware virtual machine consolidation, J. Softw. 25 (7) (2014) 1388–1402.
[2] R. Birke, L.Y. Chen, E. Smirni, Data centers in the cloud: A large scale performance study, in: 2012 IEEE 5th International Conference on Cloud Computing, (CLOUD), IEEE, 2012, pp. 336–343.
[3] A. Gandhi, M. Harchol-Balter, R. Das, et al., Optimal power allocation in server farms, ACM SIGMETRICS Perform. Eval. Rev. 37 (1) (2009) 157–168. ACM.
[4] S. Srikantaiah, A. Kansal, F. Zhao, Energy aware consolidation for cloud computing, in: International Conference on Power Aware Computing and Systems, ACM, USENIX Association, San Diego, California, 2008, pp. 10–10.
[5] A. Verma, P. Ahuja, A. Neogi, pMapper: Power and migration cost aware application placement in virtualized systems, in: ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing, Springer, Berlin, Heidelberg, 2008, pp. 243–264.
[6] E. Feller, L. Rilling, C. Morin, Energy-aware ant colony based workload placement in clouds, in: Proceedings of the 2011 IEEE/ACM 12th International Conference on Grid Computing, IEEE Computer Society, 2011, pp. 26–33.
[7] W. Voorsluys, J. Broberg, S. Venugopal, et al., Cost of virtual machine live migration in clouds, in: IEEE International Conference on Cloud Computing, Springer, Berlin, Heidelberg, 2011, pp. 254–265.
[8] A. Beloglazov, J. Abawajy, R. Buyya, Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing, Future Gener. Comput. Syst. 28 (5) (2012) 755–768.
[9] M.A. Kaaouache, S. Bouamama, Solving bin packing problem with a hybrid genetic algorithm for VM placement in cloud, Procedia Comput. Sci. 60 (2015) 1061–1069.
[10] M. Mishra, A. Sahoo, On theory of vm placement: Anomalies in existing methodologies and their mitigation using a novel vector based approach, in: 2011 IEEE International Conference on Cloud Computing, (CLOUD), IEEE, 2011, pp. 275–282.
[11] J.A. Aroca, A.F. Anta, M.A. Mosteiro, et al., Power-efficient assignment of virtual machines to PMs, Future Gener. Comput. Syst. 54 (2016) 82–94.
[12] M.R. Chowdhury, M.R. Mahmud, R.M. Rahman, Implementation and performance analysis of various VM placement strategies in CloudSim, J. Cloud Comput. 4 (1) (2015) 1–21.
[13] Z.Á. Mann, Rigorous results on the effectiveness of some heuristics for the consolidation of virtual machines in a data center, Future Gener. Comput. Syst. 51 (2015) 1–6.
[14] F. Farahnakian, T. Pahikkala, P. Liljeberg, et al., Energy aware consolidation algorithm based on K-nearest neighbor regression for data centers, in: 2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing, (UCC), IEEE, 2013, pp. 256–259.
[15] S.B. Shaw, A.K. Singh, Use of proactive and reactive hotspot detection technique to reduce the number of virtual machine migration and energy consumption in data center, Comput. Electr. Eng. 47 (2015) 241–254.
[16] A. Beloglazov, R. Buyya, Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in data centers, Concurr. Comput.: Pract. Exper. 24 (13) (2012) 1397–1420.

[17] F. Farahnakian, T. Pahikkala, P. Liljeberg, et al., Utilization prediction aware VM consolidation approach for green cloud computing, in: 2015 IEEE 8th International Conference on Cloud Computing, IEEE, 2015, pp. 381–388.
[18] F. Farahnakian, P. Liljeberg, J. Plosila, LiRCUP: Linear regression based CPU usage prediction algorithm for live migration of virtual machines in data centers, in: 2013 39th Euromicro Conference on Software Engineering and Advanced Applications, IEEE, 2013, pp. 357–364.
[19] N.T. Hieu, M. Di Francesco, A. Ylä-Jääski, Virtual machine consolidation with usage prediction for energy-efficient data centers, in: 2015 IEEE 8th International Conference on Cloud Computing, IEEE, 2015, pp. 750–757.
[20] D. Kusic, J.O. Kephart, J.E. Hanson, et al., Power and performance management of virtualized computing environments via lookahead control, Cluster Comput. 12 (1) (2009) 1–15.
[21] G. Han, W. Que, G. Jia, et al., An efficient virtual machine consolidation scheme for multimedia cloud computing, Sensors 16 (2) (2016) 246–262.
[22] S.S. Masoumzadeh, H. Hlavacs, An intelligent and adaptive threshold-based schema for energy and performance efficient dynamic VM consolidation, in: European Conference on Energy Efficiency in Large Scale Distributed Systems, Springer, Berlin, Heidelberg, 2013, pp. 85–97.
[23] S.S. Masoumzadeh, H. Hlavacs, Dynamic virtual machine consolidation: A multi agent learning approach, in: 2015 IEEE International Conference on Autonomic Computing, (ICAC), IEEE, 2015, pp. 161–162.
[24] S.S. Masoumzadeh, H. Hlavacs, A cooperative multi agent learning approach to manage PMs nodes for dynamic consolidation of virtual machines, in: 2015 IEEE Fourth Symposium on Network Cloud Computing and Applications, (NCCA), IEEE, 2015, pp. 43–50.
[25] L. Chen, H. Shen, K. Sapra, Distributed autonomous virtual resource management in datacenters using finite-Markov decision process, in: Proceedings of the ACM Symposium on Cloud Computing, ACM, 2014, pp. 1–13.
[26] S. Sohrabi, I. Moser, The effects of hotspot detection and virtual machine migration policies on energy consumption and service levels in the cloud, Procedia Comput. Sci. 51 (2015) 2794–2798.
[27] Q. Wu, F. Ishikawa, Heterogeneous virtual machine consolidation using an improved grouping genetic algorithm, in: IEEE International Conference on High Performance Computing and Communications (HPCC), the International Symposium on Cyberspace Safety and Security (CSS), the International Conference on Embedded Software and Systems (ICESS), 2015, New York, pp. 397–404.
[28] S. Joshi, S. Kaur, Cuckoo search approach for virtual machine consolidation in cloud data centre, in: 2015 International Conference on Computing, Communication & Automation, (ICCCA), IEEE, 2015, pp. 683–686.
[29] Q. Zheng, J. Li, B. Dong, et al., Multi-objective optimization algorithm based on BBO for virtual machine consolidation problem, in: 2015 IEEE 21st International Conference on Parallel and Distributed Systems, (ICPADS), IEEE, 2015, pp. 414–421.
[30] H. Li, G. Zhu, C. Cui, et al., Energy-efficient migration and consolidation algorithm of virtual machines in data centers for cloud computing, Computing 98 (3) (2016) 303–317.
[31] F. Farahnakian, A. Ashraf, T. Pahikkala, et al., Using ant colony system to consolidate vms for green cloud computing, IEEE Trans. Serv. Comput. 8 (2) (2015) 187–198.
[32] N. Fenton, M. Neilb, Making decisions: using Bayesian nets and MCDA, Knowl.-Based Syst. 14 (7) (2001) 307–325.
[33] M. Wang, X. Meng, L. Zhang, Consolidating virtual machines with dynamic bandwidth demand in data centers, in: 2011 Proceedings IEEE, INFOCOM, IEEE, 2011, pp. 71–75.
[34] M. Chen, H. Zhang, Y.Y. Su, et al., Effective VM sizing in virtualized data centers, in: 12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011) and Workshops, IEEE, 2015, pp. 594–601.
[35] H. Lin, X. Qi, S. Yang, et al., Workload-driven VM consolidation in data centers, in: Parallel and Distributed Processing Symposium (IPDPS), 2015 IEEE International Conference, IEEE, 2015, pp. 207–216.
[36] I.S. Moreno, P. Garraghan, P. Townend, et al., Analysis, modeling and simulation of workload patterns in a large-scale utility cloud, IEEE Trans. Cloud Comput. 2 (2) (2014) 208–221.
[37] L. Yu, L. Chen, Z. Cai, H. Shen, Y. Liang, Y. Pan, Stochastic load balancing for virtual resource management in datacenters, IEEE Trans. Cloud Comput. PP (99) (2016) 1–1. http://dx.doi.org/10.1109/TCC.2016.2525984.
[38] K.S. Park, V.S. Pai, CoMon: a mostly-scalable monitoring system for PlanetLab, ACM SIGOPS Oper. Syst. Rev. 40 (1) (2006) 65–74.
[39] R.N. Calheiros, R. Ranjan, A. Beloglazov, et al., CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, Softw. Pract. Exp. 41 (1) (2011) 23–50.
[40] S. Shen, V. van Beek, A. Iosup, Statistical characterization of business-critical workloads hosted in cloud datacenters, in: 2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, (CCGrid), IEEE, 2015, pp. 465–474.

Zhihua Li holds a Diploma and Ph.D. in Computer Science from Jiangnan University, Jiangsu, Wuxi, China in 2009. His research interests include network technology, parallel/distributed computing, information security, data mining, pattern recognition. He is currently faculty of the Dept. of Computer Science and Technology at the Jiangnan University.

Chengyu Yan is a M.S. student in the Dept. of Computer Science and Technology at the Jiangnan University, Jiangsu, Wuxi, China. His current research interests include cloud computing, grid computing, distributed computing.

Xinrong Yu is a M.S. student in the Dept. of Computer Science and Technology at the Jiangnan University, Jiangsu, Wuxi, China. His research interests include cloud computing, parallel computing, distributed computing.

Ning Yu received B.Sc. degree in Computer Science and M.Eng. degree in Signal Processing from Communication University of China, the M.Sc. degree in Computer Science from Southern Illinois University Carbondale, IL, USA in 2009. He is currently working toward the Ph.D. degree in computer science at Georgia State University, Atlanta, GA, USA. His research area includes Data Mining, Bioinformatics, Parallel and Cloud Computing.